Lab Exercise 4– Terraform Variables

Objective:

Learn how to define and use variables in Terraform configuration.

Prerequisites:

• Install Terraform on your machine.

Steps:

1. Create a Terraform Directory:

• Create a new directory for your Terraform project.

```
Terraform-SPCM-LAB  cd EXP-4
EXP-4  terraform init
```

2. Create a Terraform Configuration File:

• Create a file named main.tf within your project directory.

EXP-4 > main.tf > ...

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "5.31.0"
    }
  }
}
provider "aws" {
  region     = "ap-south-1"
  access_key = "AKIATG3O3S6EJ4VEH3G7"
  secret_key = "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"
}
```

3. Define Variables:

• Open a new file named variables.tf. Define variables for region, ami, and instance_type.

EXP-4 > instance.tf > resource "aws_instance" "My_Instance_3" > tags

```
resource "aws_instance" "My_Instance_1" {
  instance_type = var.instance_ty
  ami           = var.ami
  count         = var.instance_count
  tags = {
    Name = "My-UPES-Instance-1"
  }
}
resource "aws_instance" "My_Instance_2" {
  instance_type = var.instance_ty
  ami           = var.ami
  count         = var.instance_count
  tags = {
    Name = "My-UPES-Instance-2"
  }
}
resource "aws_instance" "My_Instance_3" {
  instance_type = var.instance_ty
  ami           = var.ami
  count         = var.instance_count

  tags = {
    Name = "My-UPES-Instnace-3"
  }
```

variable.tf × instance.tf

EXP-4 > variable.tf > variable "ami" > default

```
1   variable "instance_ty"{
2       type=string
3       default="t2.micro"
4   }
5   variable "instance_count" {
6     type = number
7     default = 1
8   }
9   variable "ami" {
10    type = string
11    default="ami-0d63de463e6604d0a"
12  }
```

4. Initialize and Apply:

• Run the following Terraform commands to initialize and apply the Configuration

```
● → EXP-4  terraform init

  Initializing the backend...

  Initializing provider plugins...
  - Finding hashicorp/aws versions matching "5.31.0"...
  - Installing hashicorp/aws v5.31.0...
  - Installed hashicorp/aws v5.31.0 (signed by HashiCorp)

  Terraform has created a lock file .terraform.lock.hcl to record the provider
  selections it made above. Include this file in your version control repository
  so that Terraform can guarantee to make the same selections by default when
  you run "terraform init" in the future.

  Terraform has been successfully initialized!

  You may now begin working with Terraform. Try running "terraform plan" to see
  any changes that are required for your infrastructure. All Terraform commands
  should now work.

  If you ever set or change modules or backend configuration for Terraform,
  rerun this command to reinitialize your working directory. If you forget, other
  commands will detect it and remind you to do so if necessary.
● → EXP-4  terraform validate
  Success! The configuration is valid.
```

```
● → EXP-4  terraform validate
  Success! The configuration is valid.
```

```
● → EXP-4  terraform plan

Terraform used the selected providers to generate the following
execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.My_Instance_1[0] will be created
  + resource "aws_instance" "My_Instance_1" {
      + ami                                  = "ami-0d63de463e6604d0a"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
```

```
● → EXP-4  terraform apply

Terraform used the selected providers to generate the following
execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.My_Instance_1[0] will be created
  + resource "aws_instance" "My_Instance_1" {
      + ami                                  = "ami-0d63de463e6604d0a"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
```

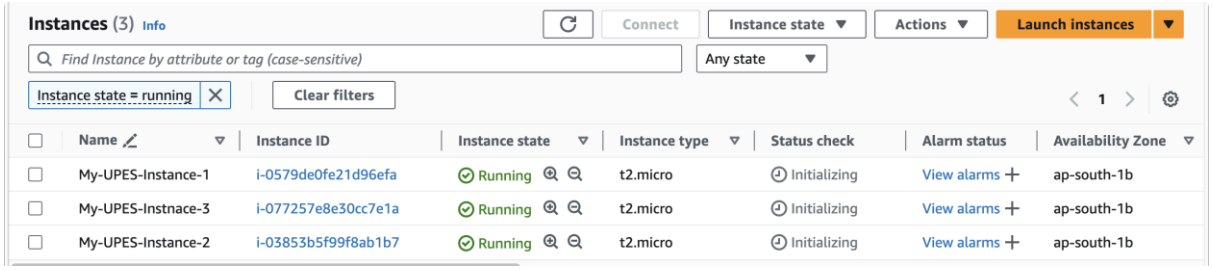
```
Plan: 3 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.My_Instance_3[0]: Creating...
aws_instance.My_Instance_1[0]: Creating...
aws_instance.My_Instance_2[0]: Creating...
aws_instance.My_Instance_1[0]: Still creating... [10s elapsed]
aws_instance.My_Instance_2[0]: Still creating... [10s elapsed]
aws_instance.My_Instance_3[0]: Still creating... [10s elapsed]
aws_instance.My_Instance_1[0]: Still creating... [20s elapsed]
aws_instance.My_Instance_3[0]: Still creating... [20s elapsed]
aws_instance.My_Instance_2[0]: Still creating... [20s elapsed]
aws_instance.My_Instance_1[0]: Creation complete after 22s [id=i-0579de0fe21d96efa]
aws_instance.My_Instance_2[0]: Creation complete after 22s [id=i-03853b5f99f8ab1b7]
aws_instance.My_Instance_3[0]: Creation complete after 22s [id=i-077257e8e30cc7e1a]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
```



5. Clean Up:

After testing, you can clean up resources.

```
→  EXP-4  terraform destroy
aws_instance.My_Instance_3[0]: Refreshing state... [id=i-077257e8e30cc7e1a]
aws_instance.My_Instance_2[0]: Refreshing state... [id=i-03853b5f99f8ab1b7]
aws_instance.My_Instance_1[0]: Refreshing state... [id=i-0579de0fe21d96efa]

Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_instance.My_Instance_1[0] will be destroyed
  - resource "aws_instance" "My_Instance_1" {
      - ami                                  = "ami-0d63de463e6604d0a" -> null
      - arn                                  = "arn:aws:ec2:ap-south-1:220886439816:instance/i-0579de0fe
d96efa" -> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "ap-south-1b" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-0579de0fe21d96efa" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t2.micro" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - monitoring                           = false -> null
      - placement_partition_number           = 0 -> null
      - primary_network_interface_id         = "eni-009f9505fae59ee61" -> null
      - private_dns                          = "ip-172-31-14-241.ap-south-1.compute.internal" -> null
```

```
Plan: 0 to add, 0 to change, 3 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_instance.My_Instance_3[0]: Destroying... [id=i-077257e8e30cc7e1a]
aws_instance.My_Instance_1[0]: Destroying... [id=i-0579de0fe21d96efa]
aws_instance.My_Instance_2[0]: Destroying... [id=i-03853b5f99f8ab1b7]
aws_instance.My_Instance_3[0]: Still destroying... [id=i-077257e8e30cc7e1a, 10s elapsed]
aws_instance.My_Instance_2[0]: Still destroying... [id=i-03853b5f99f8ab1b7, 10s elapsed]
aws_instance.My_Instance_1[0]: Still destroying... [id=i-0579de0fe21d96efa, 10s elapsed]
aws_instance.My_Instance_2[0]: Still destroying... [id=i-03853b5f99f8ab1b7, 20s elapsed]
aws_instance.My_Instance_3[0]: Still destroying... [id=i-077257e8e30cc7e1a, 20s elapsed]
aws_instance.My_Instance_1[0]: Still destroying... [id=i-0579de0fe21d96efa, 20s elapsed]
aws_instance.My_Instance_3[0]: Still destroying... [id=i-077257e8e30cc7e1a, 30s elapsed]
aws_instance.My_Instance_1[0]: Still destroying... [id=i-0579de0fe21d96efa, 30s elapsed]
aws_instance.My_Instance_2[0]: Still destroying... [id=i-03853b5f99f8ab1b7, 30s elapsed]
aws_instance.My_Instance_3[0]: Destruction complete after 31s
aws_instance.My_Instance_1[0]: Destruction complete after 31s
aws_instance.My_Instance_2[0]: Destruction complete after 31s

Destroy complete! Resources: 3 destroyed.
→  EXP-4
```